Complexity Comparison of Non-Binary LDPC Decoders

Laura Conde-Canencia, Ali Al-Ghouwayel, Emmanuel Boutillon

ICT-MobileSummit 2009 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2009
ISBN: 978-1-905824-12-0

Complexity Comparison of Non-Binary LDPC Decoders

Laura CONDE-CANENCIA, Ali AL GHOUWAYEL, Emmanuel BOUTILLON

Université de Bretagne Sud, CNRS, LabSTICC, Centre C. Huyguens
BP92116, Lorient 56100 Cedex, France
Université Européenne de Bretagne, France
Tel: +33 297874528, Fax: +33 297874560, Email: {name.surname@univ-ubs.fr}

Abstract: This paper presents a detailed complexity study of the existing non-binary LDPC decoding algorithms in order to rigorously compare them from a hardware perspective. The Belief Propagation algorithm is first considered as well as its derivative versions in the frequency and logarithm domains. We then focus on the Extended Min-Sum and its recent simplified version. For each algorithm, the number of operations in an elementary step of the check and variable nodes is determined. Finally we evaluate the interest of the application of the simplified Extended Min-Sum algorithm to a new family of non-binary LDPC codes designed in the framework of the DaVinci project[1].

Keywords: non-binary LDPC codes, iterative decoding algorithms, complexity analysis, Extended Min-Sum.

1. Introduction

The potential of LDPC codes designed over high-order finite fields GF(q) is now well-known. These codes have shown to improve the performance of binary LDPC for small and moderate codeword lengths. However, the decoder complexity increases with q, limiting the design possibilities and motivating the search of simplified decoding algorithms. We present by the following a brief overview of GF(q) LDPC decoding algorithms focusing on complexity issues.

The Belief Propagation (BP) algorithm presents computational complexity dominated by $O(q2)$ when directly applied to GF(q) LDPC codes. This is why, considering values of $q \geq 16$ results in prohibitive complexity. However, as proposed in [1] and [2], the frequency domain can be considered if $q=2p$. This FFT-Based BP decoding reduces complexity to $O(d_c.p.q)$. This algorithm was also described in the logarithm domain [3], leading to the so-called log-BP-FFT.

A reduced-complexity decoding algorithm for LDPC codes was presented in [4]. This algorithm, called the Extended Min-Sum (EMS) algorithm, is based on a generalization of the Min-Sum (MS) algorithm used for binary LDPC codes ([5], [6] and [7]). The basic idea is to use only a limited number n_m of reliabilities in the messages at the input of the check node in order to reduce the computational burden of the check node update. With $n_m << q$, the complexity of a check node varies in $O(q.\log q)$, using log-density-ratio representations

[1] This work is supported by the European FP7 ICT-STREP DAVINCI project. Contract Number: INFSO-ICT-216203

of the messages and without sacrificing too much in performance (even for high order field codes, up to GF(256)).

A new implementation of the EMS decoder has recently been proposed in [8]. A particularity of this new algorithm is that it takes into account the memory problem of the non-binary LDPC decoders, together with a significant complexity reduction per decoding iteration. The key feature of the new EMS decoder is to extend the principle of truncating the vector messages to both the check node and data node inputs. The authors truncate the messages from q to n_m values in an efficient way so as to reduce the truncation impact on the performance of the code. Moreover, they introduce an efficient offset correction to compensate for the performance loss. The complexity of the EMS decoder is now theoretically dominated by $O(n_m . \log n_m)$, with $n_m << q$, which is an important complexity reduction compared to all existing methods [3], [9], [4]. However, since parallel insertion is needed for reordering the vector messages, hardware complexity is in practice dominated by $O(n_m^2)$; and this is the complexity level that we consider for our study.

In this paper, the non-binary LDPC codes that we consider are those designed in the framework of the DaVinci project. This family of codes is defined on GF(64) and characterized by a fixed variable node degree d_v=2 and four values of check node degree d_c=4, 6, 8 and 12, leading to code rates R = 1/2, 2/3, 3/4 and 5/6.

This paper is organized as follows: Section 2 deals with the complexity of BP-like algorithms applied to GF(q) LDPC codes. Section 3 focuses on simplified decoding on the logarithm domain, considering the EMS and its simplified version. Finally, Section 4 presents the comparative study and draws conclusions on the interest of simplified EMS.

2. Complexity of BP-like decoding algorithms applied to non-binary LDPC

In this section we present the complexity in terms of number of operations at both check and variable processor nodes of the BP decoding algorithm and its derivative versions in the frequency and logarithm domains.

2.1 The BP decoding algorithm

As in the BP decoding algorithm the check node processing is a convolution of probability densities on GF(2p), the computational complexity rapidly becomes prohibitively large and the number of basic operations required to perform the parity-check node update (sum-product step) varies exponentially with both the field order q and the parity check degree d_c.

A recursive implementation of the check node update (as presented in [10]), can reduce complexity to $O(d_c q^2)$. This backward-forward calculation (same principle as in the BCJR algorithm) for a check node is represented in Figure 1 for d_c= 4. U_{pic} is the message going from permutation node i to the check node and V_{pci} is the message from the check node to permutation node i ($1< i < d_c$). Note that each $\otimes$ operator performs a direct convolution of two distributions of size q (i.e. q^2 operations).

The number of operations needed to implement an elementary step of this algorithm at the variable node and the check node level is presented in Table 1, as a function of d_c and particularized for d_v = 2.

We can conclude that, in general, any implementation of the BP algorithm for non-binary LDPC remains too complex to allow very-high order fields and q must be limited to 16 for a reasonable hardware implementation.

Table 1: Number of operations to perform an elementary step at the check node and variable node in the BP algorithm ($d_v = 2$)

	Number of additions	Number of multiplications	Number of divisions
Variable node	$q-1$	q	q
Check node	$3(d_c -2)\ q\ (q-1)$	$3(d_c -2)\ q^2$	0

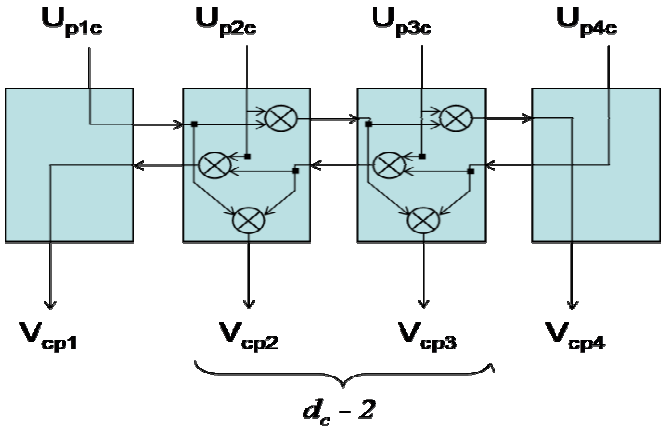

Figure 1: Check node backward-forward computation for BP decoding, $d_c = 4$

2.2 FFT-based BP decoding on GF(2^p)

In [1], the authors propose to perform the computation of the check node update in the frequency domain, which transforms the convolution into a simple product (see also [3] and [2]). This way, the computational complexity of the check node update is reduced to $O(d_c.p.q)$. Figure 22 shows the FFT and the backward-forward calculation for a check node in the FFT-BP decoding algorithm.

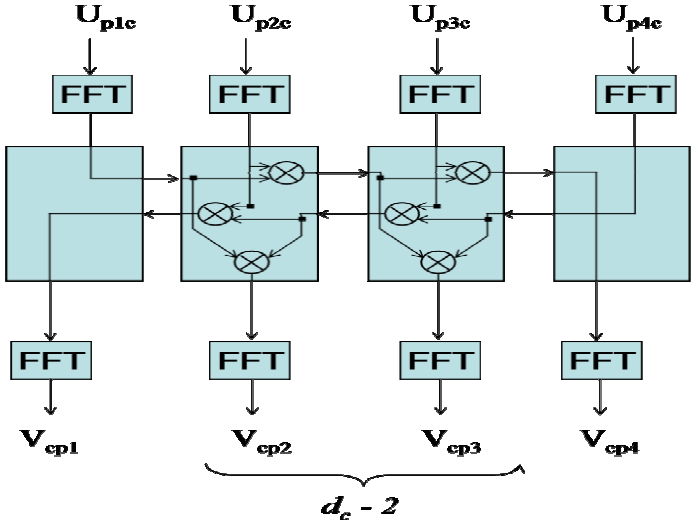

Figure 2: Check node backward-forward computation for FFT-based BP decoding, $d_c = 4$

This reduced-complexity BP algorithm allows decoding non-binary LDPC codes in very-high-order fields and for large values of d_c, or equivalently, high code rates, since $R \geq 1 - d_v / d_c$. Table 2 gives the number of operations required to perform an elementary step in the FFT BP algorithm ($p = log_2\ q$) for both variable and check nodes. Each $\otimes$ operator performs the convolution of two distributions in the frequency domain, that is q pairwise multiplications. Each FFT has a complexity of $qp/2$ multiplications and $qp/2$ additions.

Table 2: Number of operations to perform an elementary step at the check node and variable node in the FFT-BP algorithm ($d_v = 2$)

	Number of additions	Number of multiplications	Number of divisions
Variable node	$q-1$	q	q
Check node	$d_c qp$	$d_c qp + 3q(d_c-2)$	0

2.3 Decoding in the logarithm domain (log-BP)

The interest of decoding algorithms in the logarithm domain is well-known. First, it transforms the products into simple sums and the normalization of the messages is no longer required. The second reason is that log-domain algorithms are usually more robust to the quantization effects when the messages are stored on a small number of bits [9], [11]. The practical decoding algorithms for turbo-codes (max-log-map, max*-log-map) and LDPC codes (MS, logBP) are all expressed in the logarithm domain.

Table 3: Number of operations per decoding iteration in the log-BP algorithm

	Number of max*	Number of additions	Number of multiplications
Variable node	$q-1$	q	0
Check node	$3(d_c-2)q(q-1)$	$3(d_c-2)q^2$	0

Table 3 presents the number of operations needed to perform an elementary step of the variable and the check nodes. The max* operator is defined as: $\max^*(x_1, x_2) = \log(e^{x1} + e^{x2}) \approx \max(x_1, x_2)$. Note that this approximation is the one adopted for the simplified algorithms presented in the following section.

Finally, note also that the BP FFT algorithm in the logarithm domain combines the advantages of the logarithmic representation for hardware implementation and those of the frequency domain for computational complexity. An analysis of the algorithm in [9] yields that complexity per iteration is $O(Nd_vq)$ at the variable nodes and $O(Md_cq^2)$ at the check nodes, when the recursive updating of [10] is adopted.

3. Simplified decoding in the logarithm domain

The algorithms presented in the previous section are mathematically equivalent and present the same performance. The main restriction of all these is the limitation on the field order at a reasonable hardware complexity. We present in this section the EMS algorithm which represents an interesting simplified solution for reduced-complexity non-binary LDPC decoding. We focus on two implementations: the one presented in [4] and a more recent and simplified one [8].

The EMS algorithm proposed in [4] is a generalization of the MS algorithm used for binary codes, and has the advantage of performing additions only, while using only a

limited number of messages for the check node update. The significant reduction of the complexity of the check node update is due to the fact that only the n_m largest values of the messages at the input of the check node. Another point is that the EMS algorithm can be applied to both regular and irregular non-binary LDPC codes as the simplification is locally performed at the check node.

However, the sub-optimality of the EMS algorithm introduces a performance degradation compared to the BP algorithm. The main reason for this degradation is that the reliabilities computed in the EMS algorithm are overestimated. This causes the sub-optimal algorithm to converge too rapidly to a local minimum, which is most often a pseudo-codeword. This behaviour has also been observed for binary LDPC codes decoded with MS, as well as for turbo codes decoded with max*-log-map (either parallel or block turbo codes). For binary LDPC codes, a simple technique that is used to compensate for this overestimation is to reduce the magnitude of the messages at the variable-node input by means of a factor or an offset [11]. These correction techniques were applied to the EMS algorithm, either using a factor correction or an offset correction [4], and simulations showed that the EMS algorithm can approach the performance of the BP-FFT decoder, and even in some cases slightly outperform the BP-FFT decoder.

The complexity of the EMS algorithm is $O(d_c n_m q)$ per check node. For values of n_m providing near-BP error performance, this complexity is roughly the same as that of the BP-FFT decoder, but without multiplications or divisions. The EMS algorithm then becomes a good candidate for hardware implementation of non-binary LDPC decoders, because of its reduced complexity and the small or negligible performance degradation it introduces.

A new implementation of the EMS algorithm has recently been proposed in [8]. This approach aims at reducing both complexity and memory requirements of the EMS non-binary LDPC decoder. In [4], the output messages of the check node are composed of q values, thus the complexity of a single parity check node varies in $O(n_m q)$ and all the messages in the graph are stored with their full representation of q real values, implying high memory requirements. The originality of the new implementation is to store only n_m values in all vector messages (at both variable and check nodes).

As in [4], the performance degradation due to the truncation of the messages can be mitigated with a proper compensation of this information loss. The author in [8] proposes a single scalar offset as correction factor, whose value is optimized with density evolution methods.

Table 4 presents the number of operations involved in each variable and check node. In this analysis, we consider practical hardware implementation. n_{op} is the number of necessary steps so that all the n_m values of the output vector are computed. The memory requirements depend linearly on n_m and the complexity is dominated by $O(n_m^2)$. The main challenge is then to fix n_m to the minimum value that assures performance approaching the BP decoder.

Table 4: Number of operations to perform an elementary step at a variable node (with $d_v = 2$) and a check node with the simplified EMS algorithm

	Number of max	Number of real additions	Number additions in GF(q)
Variable node	n_m (n_m+2)	n_m	0
Check node	$3(d_c-2)n_{op}$ n_m	$3(d_c-2)(n_{op} + n_m)$	$3(d_c-2)(n_{op} + n_m)$

4. Complexity comparison

Tables 5 and 6 summarize the study presented in Sections 2 and 3. In Table 5 we consider the number of operations needed to perform a variable node update for the fixed variable node degree d_v=2. Note the significant complexity reduction introduced by the simplified version of the EMS compared to BP or FFT-BP: only n_m (n_m+2) Max operations and n_m additions are required (with $n_m << q$).

Table 6 considers the check node update, which typically represents the main computational load in LDPC decoders. The number of operations in an elementary step is presented for each decoding algorithm. In Figure 7 we compare the complexity of the FFT-BP and the simplified EMS algorithm for the DaVinci codes (this is q = 64 and d_c = 4, 6, 8, 12) in terms of number of additions at the elementary step of a check node update. The curves show that the EMS algorithm is only interesting for values of n_m under a certain threshold. This threshold depends on the code rate (Table 7). Simulation results show that for R=1/2 practical values of n_m are $12 < n_m < 24$. For R=5/6, we recommend n_m = 8. Note that the value of n_{op} is fixed to $2*n_m$.

Table 5: Number of operations to perform an elementary step in a variable node (with d_v = 2) for different decoding algorithms

Decoding algorithm	Number of operations in an elementary step of a **variable node**				
	Multiplications	Divisions	Max*	Max	Additions
BP	Q	q	0	0	q-1
FFT-BP	Q	q	0	0	q-1
Log-BP	0	0	q-1	0	q
Simplified EMS	0	0	0	n_m (n_m+2)	n_m

Table 6: Number of operations to perform an elementary step in a check node (with variable d_c) for different decoding algorithms

Decoding algorithm	Number of operations in an elementary step of a **check node**			
	Multiplications	Max*	Max	Additions
BP	$3(d_c -2)\, q^2$	0	0	$3(d_c -2)\, q\, (q-1)$
FFT-BP	$d_c\, qp + 3(d_c -2)q$	0	0	$d_c\, qp$
Log-BP	0	$3(d_c-2)q(q-1)$	0	$3(d_c-2)q^2$
Simplified EMS	0	0	$3(d_c-2)n_{op}\, n_m$	$3(d_c-2)(n_{op} + n_m)$ (real) $3(d_c-2)(n_{op} + n_m)$ (in GF(q))

5. Conclusion

This paper presented a state-of-the-art in non-binary LDPC decoding and a detailed study of the complexity presented by each decoding algorithm. Special attention was given to EMS and its simplified version, which introduces significant complexity reduction without nearly no performance loss. From the complexity curves we deducted the optimal values of the truncation parameter n_m for the application of the simplified EMS to a family of GF(64)-LDPC codes. The continuation of the study will focus on new issues for complexity reduction of the simplified EMS. The idea is that if complexity is even more reduced, we can increase n_m and then improve performance.

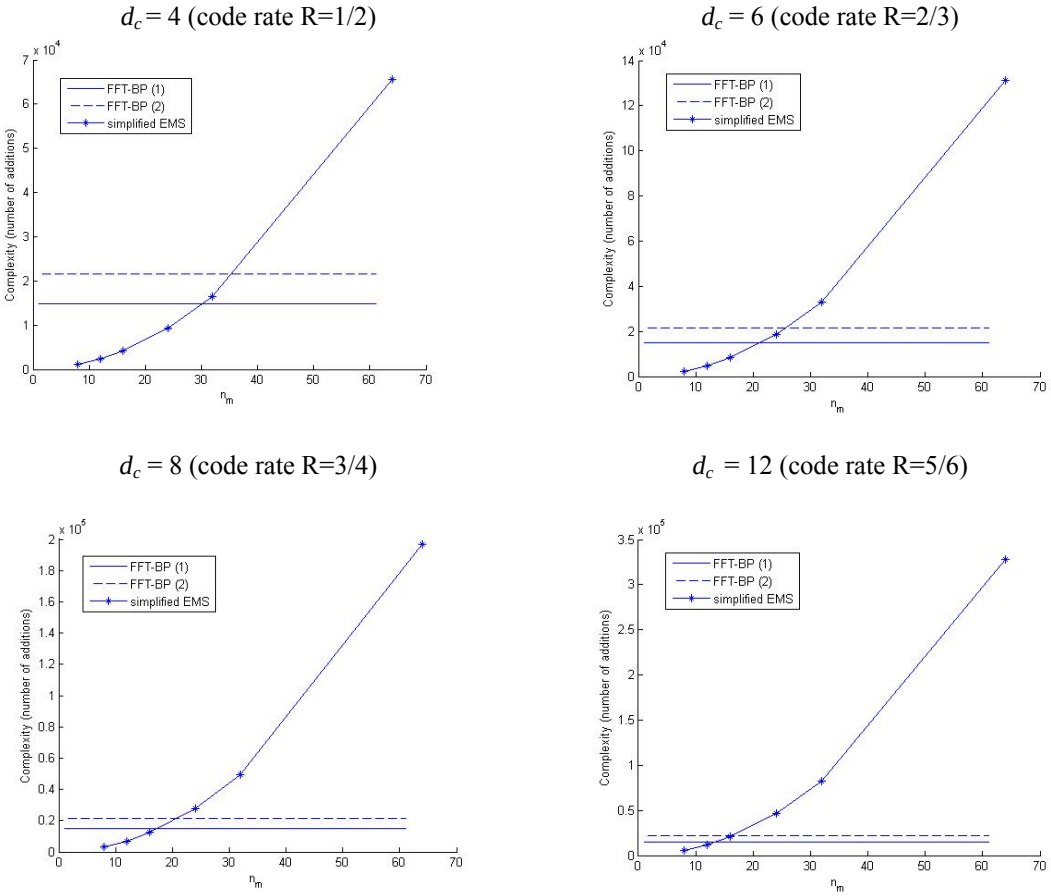

Figure 7: Check node complexity comparison in number of additions for the FFT-BP and the EMS algorithms. FFT-BP(1) stands for complexity evaluation considering that 1 multiplication equals 8 additions, idem for FFT-BP(2) 1 multiplication equals 12 additions

Table 7: Values of n_m for which the EMS algorithm becomes interesting in terms of complexity

Code rate, R	n_m values
R = ½	$n_m < 30$
R = 2/3	$n_m < 20$
R = 3/4	$n_m < 15$
R = 5/6	$n_m < 12$

References

[1] D. J. C. MacKay and M. Davey, "Evaluation of Gallager codes for short block length and high rate applications", Proc. IMA Workshop Codes, Syst., Graphical Models, 1999

[2] L. Barnault and D. Declercq, "Fast decoding algorithm for LDPC over GF", Proc. Inf. Theory Workshop Paris, France, Mar. 2003, p. 70

[3] H. Song and J. R. Cruz, "Reduced-complexity decoding of Q-ary LDPC codes for magnetic recording", IEEE Trans. Magn., vol. 39, pp. 1081-1087, Mar. 2003

[4] D. Declercq and M. Fossorier, "Decoding algorithms for Nonbinary LDPC codes over GF(q)", IEEE Trans. Comm., Vol. 55, No. 4, April 2007

[5] J. Zhao, F. Zarkeshvari and A. H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding LDPC codes", IEEE Trans. Commun., vol. 53, pp. 549, Apr. 2005

[6] M. Fossorier , M. Mihaljević and H. Imai, "Reduced complexity iterative decoding of LDPC codes based on belief propagation", IEEE Trans. Commun., vol. 47, pp. 673, May 1999.

[7] F. Kschischang, B. Frey and H.-A. Loeliger, “Factor graphs and the sum product algorithm”, IEEE Trans. Inf. Theory, vol. 47, pp. 498, Feb. 2001

[8] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, P. Urard, “Low-complexity, Low-memory EMS algorithm for non-binary LDPC codes”, in Proceedings of IEEE Int.Conf. Commun, ICC’2007, pp. 671-676, Glasglow, UK, June 2007,

[9] H. Wymeersch, H. Steendam, and M. Moeneclaey, “Log-domain decoding of LDPC codes over GF(q)”, in Proc. IEEE Int.Conf. Commun., ICC’2004, pp. 772–776, Paris, France, Jun. 2004,

[10] M. Davey and D. J. C. MacKay, “Low density parity check codes over GF(q),” IEEE Commun. Lett., vol. 2, no. 6, pp. 165–167, Jun. 1998

[11] J. Chen, A. Dholakia , E. Eleftheriou , M. Fossorier and X.-Y. Hu “Reduced-complexity decoding of LDPC codes,” IEEE Trans. Commun, vol. 53, Issue 8, pp. 1288-1299, Aug. 2005